UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/886,178	06/20/2001	David Wallman	SUN1P830/P6124	5987

22434 7590 11/30/2004

BEYER WEAVER & THOMAS LLP
P.O. BOX 778
BERKELEY, CA 94704-0778

EXAMINER	
CHOW, CHIH CHING	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 11/30/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/886,178	WALLMAN ET AL.	
	Examiner	Art Unit	
	Chih-Ching Chow	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *20 June 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-25* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-25* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on _____ is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *06/20/2001*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the application filed on June 20, 2001.

2. The priority date considered for this application is June 20, 2001.

3. Claims 1-25 have been examined.

Specification

4. The disclosure is objected to because of the following informalities: U.S. Patent application Ser. No. is missing in paragraph 001. Appropriate correction is required.

5. The use of the trademark JAVA has been noted in this application. It should be capitalized wherever it appears and be accompanied by the generic terminology.

Although the use of trademarks is permissible in patent applications, the proprietary nature of the marks should be respected and every effort made to prevent their use in any manner which might adversely affect their validity as trademarks. To expedite correction on this matter, the examiner suggests the following guidelines for Applicant to follow in amending the specification:

a. Capitalize each letter of a trademark or accompany the trademark with an appropriate designation symbol, e.g., ™ or ®, as appropriate.

b. Use each trademark as an adjective modifying a descriptive noun. For example, it would be appropriate to recite "a JAVA class file" or "a JAVA application". Note that in these examples, "class file" and "application" provide accompanying generic terminology, describing the context in which the trademark is used. By itself, the trademark JAVA specifies only the source of the so-labeled products, namely SUN Microsystems, Inc.

Claim Rejections - 35 USC § 101

6. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

7. Claims 11-19 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claim 11 merely recites a Java™ optional attribute generator, which is merely a software component, i.e., computer program per se. Such claimed matter, which is descriptive material per se, non-functional descriptive material is not statutory because it is not a physical "thing" nor a statutory process as there are not 'acts" being performed; it is not limited to "a practical application of an abstract idea which produced a useful, concrete, and tangible result." State Street Bank & Trust v. Signature Financial Group, Inc., 149 F. 3d 1368, 1375 n. 9 (Fed. Cir.

1998). Such claimed computer programs do not define any structural and functional interrelationships between the computer program and other claimed aspects of the invention, which permit the computer's program's functionality to be realized. Since a computer program is merely a set of instructions capable of being executed by a computer, the program itself is not a process, without the computer-readable medium needed to realize the computer's functionality. Thus, Applicants fail to disclose that this software component is tangibly embodied and executed by a piece of hardware and that their functions have practical applications which produce useful, concrete, and tangible results under the State Street Formulation. In contrast, a claimed computer-readable medium encoded with a computer program defines structural and functional interrelationships between the computer program and the medium which permit the computer program's functionality to be realized, and is thus mandatory. *Warmerdam*, 33 F.d at 1361, 31 USPQ 2d at 1760. In re *Sarkar*, 588 F.2d 1330, 1333, 200 USPQ 132, 137 (CCPA 1978). See MPEP §2106 (IV)(B)(1)(a).

On this basis, claim 11 is rejected under 35 U.S. C. § 101. Claims 12-19, which depend from claim 11, are also rejected under 35 U.S.C. § 101 for the same reason.

Claim Rejections - 35 USC § 112

8. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

9. Claims 1, 2, 5, 11, and 20 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite in that it fails to point out what is included or excluded by the claim language.

10. As per independent claims 1, 11, and 20 recites: "Java runtime optimization", where it is not clearly defined. Since Java has a virtual machine, which can optimize the program at runtime; therefore examiner assumes it means "**optimization information** generated by the Java program". Thus, claims 2-10, 12-19 and 21-25 are also rejected on, for being dependent on rejected base claims, respectively. Appropriate corrections are required.

11. As per independent claim 2 recites: "implemented as the last attribute in said attribute table", where it is not clearly defined. Examiner assumes it means each of the data items in the table is visited. Appropriate corrections are required.

12. As per independent claim 5 recites: "to read first, last, and next optional attributes", where it is not clearly defined. Examiner assumes it means each of the data items in the table is visited. Appropriate corrections are required.

13. Claims 1,2,4, 5-8, 11-20, 22, 23 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite in that it fails to point out and distinctly claiming the subject matter which the applicant regards as his invention.

14. Claims 1,2,4, 5-8, 11-20, 22, 23 recite the limitation "Java". JAVA, which is a bytecode programming language, is a trademark and is sued to specify only the source of the so-labeled products, namely SUN Microsystems, Inc. The presence of a trademark name in a claim is not per se improper under 112 (2nd). However, it is important to recognize that a trademark or trade name is used to identify a source of goods, and not the good themselves. Thus a trademark or trade name does not identify or describe the goods associated with the trademark or trade name.

If the trademark or trade name is used in a claim as a limitation to identify or describe a particular material or product, the claim does not comply with the requirements of the 112 (2nd) Ex Parte Simpson, 218 USPQ 1020 (Bd. App. 1982).

The claim scope is uncertain since the trademark or trade name cannot be used properly to identify any particular material or product. In fact, the value of a

trademark would be lost to the extent that it became descriptive of a product, rather than used as an identification of a source or origin of a product. Thus, the use of a trademark or trade name in a claim to identify or describe a material or product would not only render a claim indefinite, but would also constitute an improper use of the trademark or trade name.

Claim Rejections - 35 USC § 103

15. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

16. Claims 1-18, 20-25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Robert J. Cyran et al. U.S. Patent No. 6,412,107 (hereinafter "Cyran"), in view of Maynard Paul Johnson et al. U.S. Patent No. 6,330,709 (hereinafter "Johnson").

CLAIM

1. In a Java computing environment, a method of generating optional attributes in a Java class file, said method comprising:

(a) receiving as input a Java runtime optimization;

(b) generating one or more optional attributes based on said Java runtime optimization; and

(c) writing said one or more optional attributes in an attribute table portion of a Java class file.

Cyran / Johnson

For items (a) and (b), in Cyran, refer to FIG.4, and see column 6, lines 16-20, "operation continues at decision block 56 where the user may choose to **perform optimization analysis** on the **input code** *(receiving input Java code)*. If **selected** (*optional*), operation continues to block 58 where the code analyzer 36 is invoked to pre-process, i.e., pre-compile the **input code**." Further, in Cyran column 4, lines 48-55, "the presence of this **attribute in the class file** 14 informs the **Java runtime system** to JIT compile the intermediate codes for the method that includes the 'Code' **attribute**. In addition, the COM.TexasInstruments.JIT **attribute** is also used to pass **optimization information** generated by the code preparation system 12 or directions to the JIT compiler instructing it to perform its own **optimizations at compile time**, as discussed hereinabove." For item (c), in Cyran column 2, lines 51-54, "The **input code** 11 may also be source code, such as **Java source code**, which is first translated into intermediate code format before being analyzed. The **optimization information** it provided as additional **attributes** added to **class files** 14 generated by the code preparation system 12." Cyran teaches all aspects of claim 1, but he does not mention 'attribute table' specifically, however, Johnson teaches it in an analogous prior art. In Johnson, column 5, lines 28-32, "Another

approach which provides object persistence is 'externalization'. Externalization is the means or protocol used in object-oriented programming for transferring data out of an object. In essence **the 'state data' that defines the attributes of an object** are "externalized", or written out of the object, **into a different format that is easily stored in the local data store.** (*attribute table*)" Also, in column 11, lines 53-67, "When the virtual address translator 210 receives an SAS address to translate it runes the SAS address through hasher 215 to generate a key number n. The hash **table** 216, preferably a list of pointers to page **table** entry lists, is then searched to locate the page **table** entry list in the lookaside buffer 218 corresponding to that key number n. The page **table** entry list corresponding to that key number n in the lookaside buffer 218 is then searched to determine if requested data is in the page cache 212." It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement Cyran's disclosure of the optimization Java environment by writing the attributes to an attribute table taught by Johnson, for the purpose of retrieving for data object from the storage (Johnson, column 12, lines 6-7).

2. A method as recited claim 1, wherein said one or more optional

For the feature of claim 1 see claim 1 rejection. Cyran teaches all the aspects

attributes are implemented as the last attribute in said attribute table portion of said Java class file.

of this claim except 'the last attribute' part. However, Johnson teaches that feature, in column 14, lines 48-55, "Java class information such as methods, fields and constant pool data is loaded into Class Encapsulator object by the Persistent ClassLoader object in the Persistent Container object, where it can be **retrieved** as needed. Java object interaction such as field data **reads and writes** are performed on the Object Encapsulator object. The Persistent Handle object is created to persistently store SAS address pointers to the Object Encapsulator object and its corresponding Class Encapsulator object for **each** persistent object" (see 112 (2) rejection, item 11, above).
It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement Cyran's disclosure of the optimization Java environment by writing the attributes to an attribute table and retrieve it as needed taught by Johnson, for the purpose of retrieving for data object from the storage (Johnson, column 14, lines 50-51)

3. A method as recited claim 1, wherein said method further comprises:
generating computer program code that implements an application programming interface suitable for loading said one or more optional

For the feature of claim 1 see claim 1 rejection. In Cyran, column 6, lines 3-5, "The code preparation system (for *computer program code*) of the present invention provides developers with an **interface** that allows them to **select** (*loading*) these individual optimizations."

attributes.

4. A method as recited claim 3, wherein said application programming interface can be used to read said one or more optional attributes from said Java class file.	For the feature of claim 3 see claim 3 rejection. See claim 2 rejection (*retrieving data object*), further in Johnson, column 4, lines 56-58, "the only way to **retrieve** *(read),* process or otherwise operate on the object is through the methods defined on the object."
5. A method as recited claim 4, wherein said application programming interface includes functions that can be used to read first, last, and next optional attributes in said Java class file.	For the feature of claim 4 see claim 4 rejection. See claim 2 rejection, where the prior art visit every object in the table; which includes **first, last** and **next** object in the table.
6. A method as recited claim 4, wherein said application programming interface includes a function suitable for finding an optional attribute in said Java class file.	For the feature of claim 4 see claim 4 rejection. See claim 3 rejection, where the prior art teaches to 'select' an object, which implies 'finding' it.
7. A method as recited claim 1, wherein said Java runtime optimization is stored in a database.	For the feature of claim 1 see claim 1 rejection. See Cyran FIG. 2, where 'storage device' can be a database, also in claim 1 (c) rejection, the attributes are stored in a 'table' which can also be a database (see Johnson FIG. 2).
8. A method as recited in claim 7, wherein said database is generated by a compiler extension or a software tool suitable for analyzing a Java application.	For the feature of claim 7 see claim 7 rejection. See Cyran FIG. 1, the Extended Class File is an extension of the Code Preparation System, which is suitable for analyzing a Java application. Also see

	Johnson FIG. 2, the Storage System is an extension of the Java Virtual Machine (*compiler*), which is also suitable for analyzing a Java application.
9. A method as recited in claim 7, wherein said database is stored in a runtime performance manager that can interact with software modules that generate and load said one or more optional attributes.	For the feature of claim 7 see claim 7 rejection. See Johnson column 2, lines 33-38, "When a process needs to access a persistent object, the process must contact the **file manager** which locates the persistent object data in a file on backing store and move a copy of the persistent object data into a memory buffer. The persistent object data must then be **reconstructed** (*generate and load*) into a persistent object in memory."
10. A method as recited in claim 7, wherein said method further comprises: updating said database to reflect generation of said one or more optional attributes.	For the feature of claim 7 see claim 7 rejection. See Johnson, column 2, lines 22-23, "The file manager **stores** (*updating*) **and retrieves data** from the permanent storage devices in the form of files."
11. In a Java computing environment, a Java optional attribute generator suitable for generation of optional attributes in a Java class file, said Java optional attribute generator operating to: receive as input a Java runtime optimization; generate one or more optional attributes based on said Java runtime optimization; and	Same as claim 1 rejection; the 'code preparation system' disclosed in Cyran's prior art functions as an 'attribute generator'.

write said one or more optional attributes in an attribute table portion of a Java class file.

12. A Java optional attribute generator as recited in claim 11, wherein said Java optional attribute generator operates to generate computer program code that implements an application programming interface suitable for loading said one or more optional attributes.	For the feature of claim 11 see claim 11 rejection. For the rest of the feature see claim 3 rejection.
13. A Java optional attribute generator as recited in claim 11, wherein an application programming interface can be used to read said one or more optional attributes from said Java class file.	For the feature of claim 11 see claim 11 rejection. For the rest of the feature see claim 4 rejection.
14. A Java optional attribute generator as recited in claim 11, wherein said Java runtime optimization is stored in a database.	For the feature of claim 11 see claim 11 rejection. For the rest of the feature see claim 7 rejection.
15. A Java optional attribute generator as recited in claim 11, wherein said database is generated by a compiler extension or a software tool suitable for analyzing a Java application.	For the feature of claim 11 see claim 11 rejection. For the rest of the feature see claim 8 rejection.
16. A Java optional attribute generator as recited in claim 11, wherein said database is stored in a runtime performance manager that can	For the feature of claim 11 see claim 11 rejection. For the rest of the feature see claim 9 rejection.

interact with software modules that generate and load said one or more optional attributes.

17. A Java optional attribute generator as recited in claim 11, wherein said optional attribute generator operates to update said database to reflect generation of said one or more optional attributes.

For the feature of claim 11 see claim 11 rejection. For the rest of the feature see claim 10 rejection.

18. A Java optional attribute generator as recited in claim 11, wherein said optional attribute generator operates to generate a description of an optional attribute.

For the feature of claim 11 see claim 11 rejection. In Cyran, column 8, lines 57-64, "The code preparation system 12 also passes **directions** (*description*) to the code interpretive runtime system in addition to passing optimization information. These **directions** are usually relating to optimizations that must be done on generated native code as opposed to being done on intermediate code. Thus, for example, **directions** can be given to the JIT compiler to perform **instruction** scheduling and peephole optimizations as described hereinabove in the **description** of the *COM*.TexasInstruments.JITOptimizations attribute."

20. A computer readable medium including computer program code for generating optional attributes in a Java class file, said computer readable medium comprising:
(a) computer program code for receiving as input a Java runtime optimization;

Same as claim 1 rejection; a readable medium is disclosed in Cyran's prior art, see FIG. 2.

optimization;
(b) computer program code for generating one or more optional attributes based on said Java runtime optimization; and
(c) computer program code for writing said one or more optional attributes in an attribute table portion of a Java class file.

21. A computer readable medium as recited in claim 20, wherein said method further comprises:
generating computer program code that implements an application programming interface suitable for loading said one or more optional attributes.

For the feature of claim 20 see claim 20 rejection. For rest of the features see claim 3 rejection.

22. A computer readable medium as recited in claim 21, wherein said Java runtime optimization is stored in a database.

For the feature of claim 21 see claim 21 rejection. For rest of the features see claim 7 rejection.

23. A computer readable medium as recited in claim 22, wherein said database is generated by a compiler extension or a software tool suitable for analyzing a Java application.

For the feature of claim 22 see claim 22 rejection. For rest of the features see claim 8 rejection.

24. A computer readable medium as recited in claim 22, wherein said database is stored in a runtime performance manager that can interact with software modules that generate and load said one or more

For the feature of claim 22 see claim 22 rejection. For rest of the features see claim 9 rejection.

optional attributes.

25. A computer readable medium as recited in claim 24, wherein said method further comprises: updating said database to reflect generation of said one or more optional attributes.	For the feature of claim 24 see claim 24 rejection. For rest of the features see claim 10 rejection.

17. Claim 19 is rejected under 35 U.S.C. 103(a) as being unpatentable over Robert J. Cyran et al. U.S. Patent No. 6,412,107 (hereinafter "Cyran"), in view of Maynard Paul Johnson et al. U.S. Patent No. 6,330,709 (hereinafter "Johnson"), and further in view of Cary Lee Bates et al. U.S. Patent No. 6,769,015 (hereinafter "Bates").

CLAIM	Cyran / Johnson / Bates
19. A Java optional attribute generator as recited in claim 18, wherein said description is in XML format.	For the feature of claim 18 see claim 18 rejection. Cyran and Johnson teach all aspects of claim 19, but he does not mention 'in XML' specifically, however, Bates teaches it in an analogous prior art. In Bates' column 6, lines 22-23, "The attributes may be encoded using, e.g., HTML or XML (extensible markup language)". It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement Cyran and Johnson's disclosure of the Attributes Generator in a Java environment by using XML taught by Bates, for the purpose of retrieving the attribute information (Bates column 6, lines 21).

Conclusion

The following summarizes the status of the claims:

35 USC § 101 claim rejection: 11-19

35 USC § 112 (2nd paragraph) rejection: 1-25

35 USC § 103 claim rejection: 1-25

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Chih-Ching Chow whose telephone number is 571-272-3693. The examiner can normally be reached on 7:00am - 3:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Chih-Ching Chow
Examiner
Art Unit 2122

CC

ANTONY NGUYEN-BA
PRIMARY EXAMINER